What is claimed is:

1. A method for controlling the computational resources of at least one coprocessor in a host computing system having a host processor, comprising

controlling the at least one coprocessor of the computing system with command buffers submitted to the at least one coprocessor by a host processor of the host computing system;

transmitting, by the at least one coprocessor, data back to the host computing system in response to commands in at least one command buffer of the command buffers; and

scheduling the transmission of the command buffers by a managing object included in the host computing system,

wherein the computational resources of the at least one coprocessor are simultaneously available to a plurality of applications instantiated on the host computing system.

2. A method according to claim 1, wherein said scheduling includes scheduling the transmission of the command buffers by an operating system included in the host computing system.

3. A method according to claim 1, wherein the managing object is notified by a coprocessor that a command buffer has finished execution.

4. A method according to claim 1, further including queuing a new command buffer for a coprocessor to begin executing when a current command buffer is finished.

5. A method according to claim 1, further including specifying a coprocessor context switch when a command buffer is submitted.

6. A method according to claim 1, wherein said managing object allows a plurality of types of coprocessor context.

7. A method according to claim 6, further including affiliating coprocessor context with a

host processor thread context.

8. A method according to claim 7, further including integrating by the managing object the context switching code for the host processor and the coprocessor.

9. A method according to claim 1, further including notifying the managing object by a coprocessor that a command buffer is invalid.

10. A method according to claim 1, further including resetting a coprocessor of the at least one coprocessor if the coprocessor is unresponsive for a predetermined period of time.

11. A method according to claim 1, further including translating by a hardware-specific driver object, via an application programming interface of the managing object, instructions of a command buffer into hardware-specific instructions during composition of the command buffer.

12. A method according to claim 11, wherein said translating runs in user mode.

13. A method according to claim 12, further including allocating a guard page at the end of the command buffer to facilitate efficient detection of buffer overflow.

14. A method according to claim 12, wherein the user mode driver and corresponding runtime component are provided in intermediate language form and the method further includes just in time (JIT) compiling on a client device having the user mode driver and runtime component.

15. A method according to claim 14, wherein the application is also provided in intermediate language form and said JIT compiling includes JIT compiling the application on the client device with the user mode driver and runtime.

16. A method according to claim 12, wherein said driver object coordinates with a corresponding kernel mode driver object to edit the command buffer before submission to hardware.

17. A method according to claim 1, wherein the at least one coprocessor includes at least one graphics processing unit.

18. A method according to claim 1, further including preempting by the at least one coprocessor upon the occurrence of an external event.

19. A method according to claim 18, wherein the external event is the operating system making a call to a corresponding kernel mode driver object to preempt the at least one coprocessor.

20. A method according to claim 18, wherein the host processor is interrupted to coordinate scheduling of processing time.

21. A method according to claim 1, further including virtualizing by the managing object at least one resource of the at least one coprocessor during editing of the control data streams of a command buffer before submission to a coprocessor.

22. A method according to claim 21, wherein the at least one resource virtualized by the managing object of the at least one coprocessor is memory.

23. A method according to claim 1, wherein the managing object uses thread synchronization primitives to coordinate the construction, scheduling and submission of coprocessor command buffers.

24. A computer readable medium having stored thereon a plurality of computer-executable

instructions for performing the method of claim 1.

25. A modulated data signal carrying computer executable instructions for performing the method of claim 1.

26. A computing device comprising means for performing the method of claim 1.

27. At least one computer readable medium having stored thereon a plurality of computer-executable modules for controlling the computational resources of at least one coprocessor in a host computing system having a host processor, the computer executable modules comprising:

a managing object for controlling the at least one coprocessor of the computing system with command buffers submitted to the at least one coprocessor by a host processor of the host computing system and for scheduling the transmission of the command buffers; and

means for transmitting, by the at least one coprocessor, data back to the host computing system in response to commands in at least one command buffer of the command buffers;

whereby the computational resources of the at least one coprocessor are simultaneously available to a plurality of applications instantiated on the host computing system.

28. At least one computer readable medium according to claim 27, wherein said managing object is included in the operating system of the host computing system.

29. At least one computer readable medium according to claim 27, wherein the managing object is notified by a coprocessor that a command buffer has finished execution.

30. At least one computer readable medium according to claim 27, wherein said managing object queues a new command buffer for a coprocessor to begin executing when a current command buffer is finished.

31. At least one computer readable medium according to claim 27, wherein said managing

object specifies a coprocessor context switch when a command buffer is submitted.

32. At least one computer readable medium according to claim 27, wherein said managing object allows a plurality of types of coprocessor context.

33. At least one computer readable medium according to claim 32, wherein said managing object affiliates coprocessor context with a host processor thread context.

34. At least one computer readable medium according to claim 33, wherein said managing object integrates the context switching code for the host processor and the coprocessor.

35. At least one computer readable medium according to claim 27, wherein a coprocessor comprises means for notifying the managing object that a command buffer is invalid.

36. At least one computer readable medium according to claim 27, wherein said managing object resets a coprocessor of the at least one coprocessor if the coprocessor is unresponsive for a predetermined period of time.

37. At least one computer readable medium according to claim 27, wherein said managing object enables translating by a hardware-specific driver object instructions of a command buffer into hardware-specific instructions during composition of the command buffer.

38. At least one computer readable medium according to claim 37, wherein said translating by the driver object runs in user mode.

39. At least one computer readable medium according to claim 38, wherein said managing object allocates a guard page at the end of the command buffer to facilitate efficient detection of buffer overflow.

40. At least one computer readable medium according to claim 38, wherein the user mode driver and corresponding runtime component are provided in intermediate language form and the user mode driver and runtime component are just in time (JIT) compiled.

41. At least one computer readable medium according to claim 40, wherein the application is also provided in intermediate language form and the application, the user mode driver and runtime are JIT compiled.

42. At least one computer readable medium according to claim 38, wherein said driver object coordinates with a corresponding kernel mode driver object to edit the command buffer before submission to hardware.

43. At least one computer readable medium according to claim 27, wherein the at least one coprocessor includes at least one graphics processing unit.

44. At least one computer readable medium according to claim 27, wherein the managing object is preempted by the at least one coprocessor upon the occurrence of an external event.

45. At least one computer readable medium according to claim 44, wherein the external event is the operating system making a call to a corresponding kernel mode driver object to preempt the at least one coprocessor.

46. At least one computer readable medium according to claim 27, wherein the host processor is interrupted to coordinate scheduling of processing time by the managing object.

47. At least one computer readable medium according to claim 27, wherein the managing object virtualizes at least one resource of the at least one coprocessor during editing of the control data streams of a command buffer before submission to a coprocessor.

48. At least one computer readable medium according to claim 47, wherein the at least one resource virtualized by the managing object of the at least one coprocessor is memory.

49. At least one computer readable medium according to claim 27, wherein the managing object uses thread synchronization primitives to coordinate the construction, scheduling and submission of coprocessor command buffers.

50. A modulated data signal carrying computer executable instructions output as a result of the execution of the plurality of computer-executable instructions of the computer readable medium of claim 27.

51. A computing device comprising means for carrying out the plurality of computer-executable instructions of the computer readable medium of claim 27.

52. A computing device for controlling the computational resources of at least one coprocessor in a host computing system having a host processor, comprising:

a managing object for controlling the at least one coprocessor of the computing system with command buffers submitted to the at least one coprocessor by a host processor of the host computing system and for scheduling the transmission of the command buffers; and

means for transmitting, by the at least one coprocessor, data back to the host computing system in response to commands in at least one command buffer of the command buffers;

whereby the computational resources of the at least one coprocessor are simultaneously available to a plurality of applications instantiated on the host computing system.

53. A computing device according to claim 52, wherein said managing object is included in the operating system of the host computing system.

54. A computing device according to claim 52, wherein the managing object is notified by a coprocessor that a command buffer has finished execution.

55. A computing device according to claim 52, wherein said managing object queues a new command buffer for a coprocessor to begin executing when a current command buffer is finished.

56. A computing device according to claim 52, wherein said managing object specifies a coprocessor context switch when a command buffer is submitted.

57. A computing device according to claim 52, wherein said managing object allows a plurality of types of coprocessor context.

58. A computing device according to claim 57, wherein said managing object affiliates coprocessor context with a host processor thread context.

59. A computing device according to claim 58, wherein said managing object integrates the context switching code for the host processor and the coprocessor.

60. A computing device according to claim 52, wherein a coprocessor comprises means for notifying the managing object that a command buffer is invalid.

61. A computing device according to claim 52, wherein said managing object resets a coprocessor of the at least one coprocessor if the coprocessor is unresponsive for a predetermined period of time.

62. A computing device according to claim 52, wherein said managing object enables translating by a hardware-specific driver object instructions of a command buffer into hardware-specific instructions during composition of the command buffer.

63. A computing device according to claim 62, wherein said translating by the driver object

runs in user mode.

64. A computing device according to claim 63, wherein said managing object allocates a guard page at the end of the command buffer to facilitate efficient detection of buffer overflow.

65. A computing device according to claim 63, wherein the user mode driver and corresponding runtime component are provided in intermediate language form and the user mode driver and runtime component are just in time (JIT) compiled.

66. A computing device according to claim 65, wherein the application is also provided in intermediate language form and the application, the user mode driver and runtime are JIT compiled.

67. A computing device according to claim 63, wherein said driver object coordinates with a corresponding kernel mode driver object to edit the command buffer before submission to hardware.

68. A computing device according to claim 52, wherein the at least one coprocessor includes at least one graphics processing unit.

69. A computing device according to claim 52, wherein the managing object is preempted by the at least one coprocessor upon the occurrence of an external event.

70. A computing device according to claim 69, wherein the external event is the operating system making a call to a corresponding kernel mode driver object to preempt the at least one coprocessor.

71. A computing device according to claim 52, wherein the host processor is interrupted to coordinate scheduling of processing time by the managing object.

72. A computing device according to claim 52, wherein the managing object virtualizes at least one resource of the at least one coprocessor during editing of the control data streams of a command buffer before submission to a coprocessor.

73. A computing device according to claim 72, wherein the at least one resource virtualized by the managing object of the at least one coprocessor is memory.

74. A computing device according to claim 52, wherein the managing object uses thread synchronization primitives to coordinate the construction, scheduling and submission of coprocessor command buffers.